

STAGE 5

Obstacle-Avoiding Robot

An autonomous robot that scans for obstacles with an ultrasonic sensor and steers around them.



Skill level: Advanced

Recommended age: 14+

Hands-on builds: 4

What you'll build

This manual walks through assembling and programming the **Stage 5: Obstacle-Avoiding Robot** step by step, alongside a full materials list and the original circuit diagram. It's a companion to — not a replacement for — the kit's original build guide, which is linked from the product page for reference.

#	Project	What it teaches
1	Ultrasonic distance test	Wire the ultrasonic sensor to the Pico and print live distance readings to confirm it's working.
2	Servo scanning mount	Attach the sensor to the SG90 servo and write code to sweep it left and right.
3	Motor driver wiring	Wire all 4 BO motors to the L298N driver for full directional control.
4	Autonomous obstacle avoidance	Combine distance sensing and servo scanning into the full decision loop that steers the robot around obstacles.

What's in the box

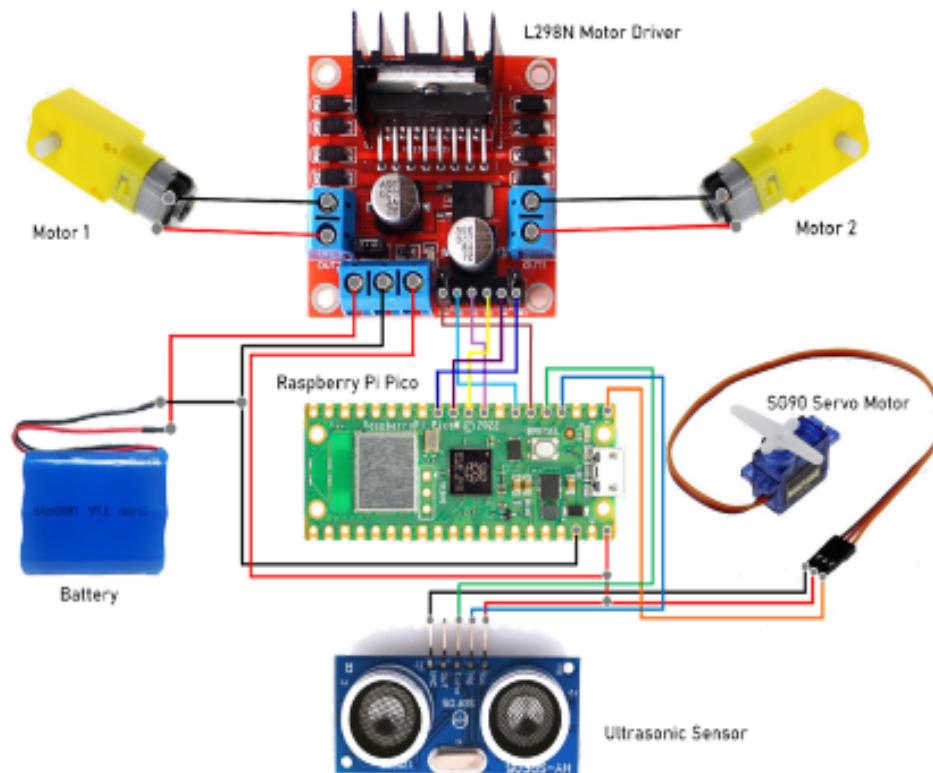
12 line items. Check every part against this list before you start building.

Item	Qty	What it's for
Raspberry Pi Pico	1	The microcontroller — runs the MicroPython program that senses and steers.
L298N motor driver	1	Lets the Pico safely switch the higher-current drive motors.
Ultrasonic sensor	1	Measures distance to the nearest object using an ultrasonic echo, like sonar.
300 RPM BO motor-straight	4	Geared DC motors that drive the wheels.
65mm robot wheel	4	Larger wheels for smoother movement.
Robot chassis	1	The frame every component mounts to.
Jumper wires	40	Wire the sensor, servo, and driver board together.
18650 Li-ion 2500mAh battery pack	1	Power source for the motors and electronics.
Breadboard	1	Prototyping surface for the sensor wiring.
SG90 servo motor	1	Rotates the ultrasonic sensor left and right to scan for the clearest path.
Data cable (type B)	1	Connects the Pico to a computer to upload the MicroPython code.
Double-sided foam tape	1	Secures the breadboard and sensor mount to the chassis.

Step-by-step build instructions

1	Assemble the chassis Mount all 4 BO motors and wheels onto the robot chassis.
2	Wire the motor driver Mount the L298N to the chassis and wire all 4 motors to its output terminals as shown in the circuit diagram on page 4.
3	Wire the driver to the Pico Connect the L298N's ENA, IN1-IN4, and ENB pins to the Pico's GPIO pins as shown in the diagram.
4	Mount the servo and sensor Fix the SG90 servo to the front of the chassis, then attach the ultrasonic sensor to the servo horn so the whole sensor rotates with it.
5	Wire the sensor and servo Connect the ultrasonic sensor's Trig and Echo pins and the servo's signal wire to the Pico's GPIO pins as shown in the diagram.
6	Connect power Wire the battery pack in and confirm the Pico, L298N, servo, and sensor all share a common ground.
7	Upload the code Flash the MicroPython code on page 6 onto the Pico as main.py.
8	Centre the servo Power on the robot and confirm the servo returns to its centre (forward-facing) position on startup.
9	Test obstacle avoidance Place an obstacle in front of the robot. It should stop, scan left and right, measure the distance each way, and steer toward the clearer side. Adjust the distance threshold in the code (the '<10' check) if it reacts too early or too late.

Circuit diagram



MicroPython code

Copy this onto the Raspberry Pi Pico as main.py using Thonny (or your preferred MicroPython IDE), then run it.

```
# Include the library files
from machine import Pin, PWM
import time

servo = PWM(Pin(0)) # Servo motor pin
servo.freq(50)      # Set the frequency

Trig = Pin(2, Pin.OUT)
Echo = Pin(3, Pin.IN)

# Motor driver pins
ENA = PWM(Pin(4))
IN1 = Pin(5, Pin.OUT)
IN2 = Pin(6, Pin.OUT)
IN3 = Pin(7, Pin.OUT)
IN4 = Pin(8, Pin.OUT)
ENB = PWM(Pin(9))
ENA.freq(1000)
ENB.freq(1000)

speed = 30000 # Speed of this robot

def forward():
    ENA.duty_u16(speed); IN1.value(0); IN2.value(1)
    ENB.duty_u16(speed); IN3.value(1); IN4.value(0)

def backward():
    ENA.duty_u16(speed); IN1.value(1); IN2.value(0)
    ENB.duty_u16(speed); IN3.value(0); IN4.value(1)

def left():
    ENA.duty_u16(speed); IN1.value(1); IN2.value(0)
    ENB.duty_u16(speed); IN3.value(1); IN4.value(0)

def right():
    ENA.duty_u16(speed); IN1.value(0); IN2.value(1)
    ENB.duty_u16(speed); IN3.value(0); IN4.value(1)

def stop():
    ENA.duty_u16(0); IN1.value(0); IN2.value(0)
    ENB.duty_u16(0); IN3.value(0); IN4.value(0)

def distance():
    Trig.value(0); time.sleep_us(4)
    Trig.value(1); time.sleep_us(10)
    Trig.value(0)
    while Echo.value() == 0:
        low = time.ticks_us()
    while Echo.value() == 1:
        high = time.ticks_us()
    t = high - low
    cm = t / 29 / 2 # Time converted to cm
    return cm

def servoLeft():
    servo.duty_u16(7000)

def servoRight():
    servo.duty_u16(3000)

def servoStart():
    servo.duty_u16(5400)

while True:
    dis = distance()
```

```
if dis < 10:
    stop(); time.sleep(1)
    servoLeft(); time.sleep(1)
    leftDis = distance(); time.sleep(0.5)
    servoStart(); time.sleep(1)
    servoRight(); time.sleep(1)
    rightDis = distance(); time.sleep(0.5)
    servoStart(); time.sleep(1)
    if leftDis > rightDis:
        left(); time.sleep(0.5); stop(); time.sleep(1)
    elif leftDis < rightDis:
        right(); time.sleep(0.5); stop(); time.sleep(1)
else:
    forward()
```

Troubleshooting

Symptom	Fix
A motor spins the wrong way	Swap its two wire connections at the driver board or switch.
Nothing powers on	Check battery orientation first, then confirm every switch or connector is fully seated.
Bluetooth won't pair (Stage 3)	Confirm the HC-05's LED is blinking (pairing mode) and try the default PIN, 1234 or 0000.
Pico isn't recognised by your computer	Hold the BOOTSEL button on the Pico while plugging in the USB cable, then release it once connected.
Line follower loses the line on curves (Stage 4)	Lower the sensor height slightly, slow the motor speed, or widen the gap between the two sensors.
Ultrasonic sensor gives erratic readings (Stage 5)	Make sure nothing else is close enough to cause echo interference, and double-check the Trig/Echo pins aren't swapped.

Need more help? Email our STEM instructors or visit the FAQ at streamedutech.com. The original source build guide for this kit (with the manufacturer's own circuit reference) is linked from this kit's product page alongside this manual.